

Teil 9



Ontologien und Beschreibungslogik

Ontologien - Einführung

➤ Ontologie

⇒ ist eine philosophische Disziplin, Zweig der Philosophie welcher sich mit der Natur und der Organisation von Wirklichkeit beschäftigt

➤ Lehre des Seins (Aristoteles, Metaphysik, IV, 1)

➤ Versucht folgende Fragen zu beantworten

✧ Was ist Sein?

✧ Was sind alle gemeinsamen Eigenschaften des Seins?

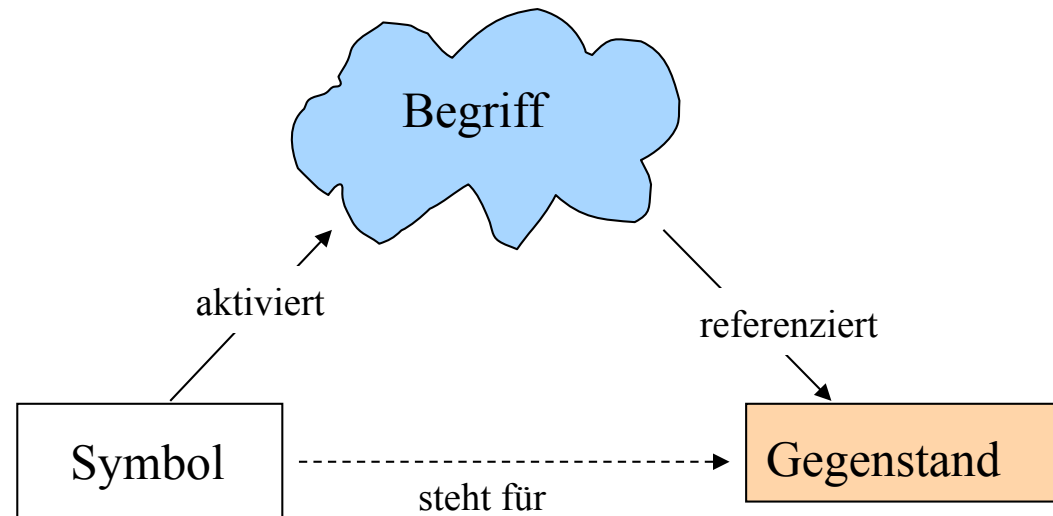
- An ontology is an explicit specification of a conceptualization. [Gruber, 93]
- An ontology is a shared understanding of some domain of interest. [Uschold, Gruninger, 96]
- Es gibt viele weitere Definitionen, in unserem Sinne ist – zunächst informell betrachtet - alles eine Ontologie was
 - ⇒ eine Konzeptualisierung eines Weltausschnitts darstellt,
 - ⇒ eine formale Spezifikation besitzt,
 - ⇒ Konsens einer Gruppe von Personen ist.

Gemeinsame Sprache

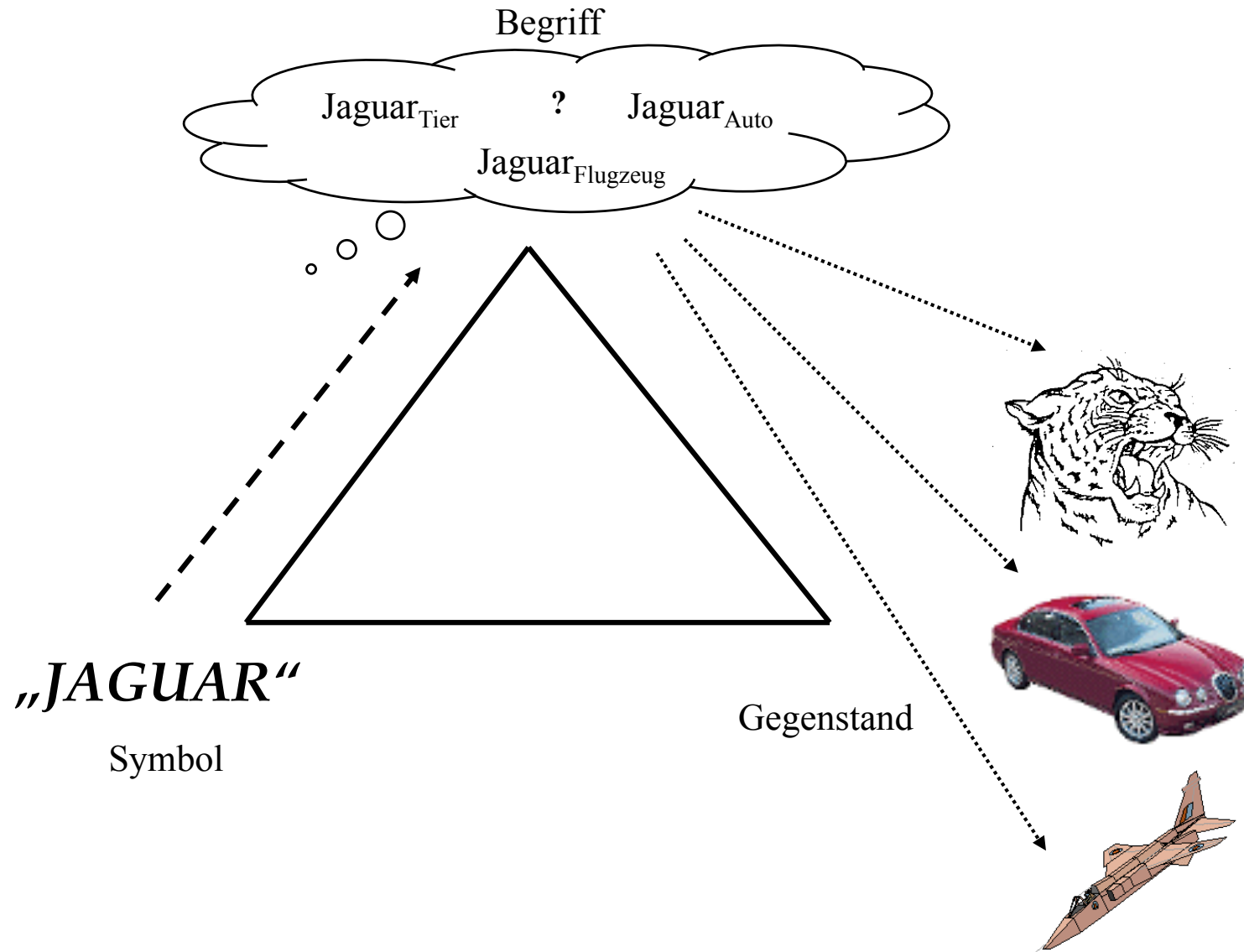
- "People can't share knowledge if they don't speak a common language,, [T. Davenport]
- "Gemeinsame Sprache" notwendig für funktionierende Kommunikation zwischen Personen bzw. Maschinen
 - ⇒ wohldefiniertes Vokabular an Symbolen (Lexemen, lexical entries)
 - ⇒ einheitliches Verständnis, welche Begriffe (concepts) und Beziehungen (relations) durch die Symbole referenziert werden

Definition von Ontologien

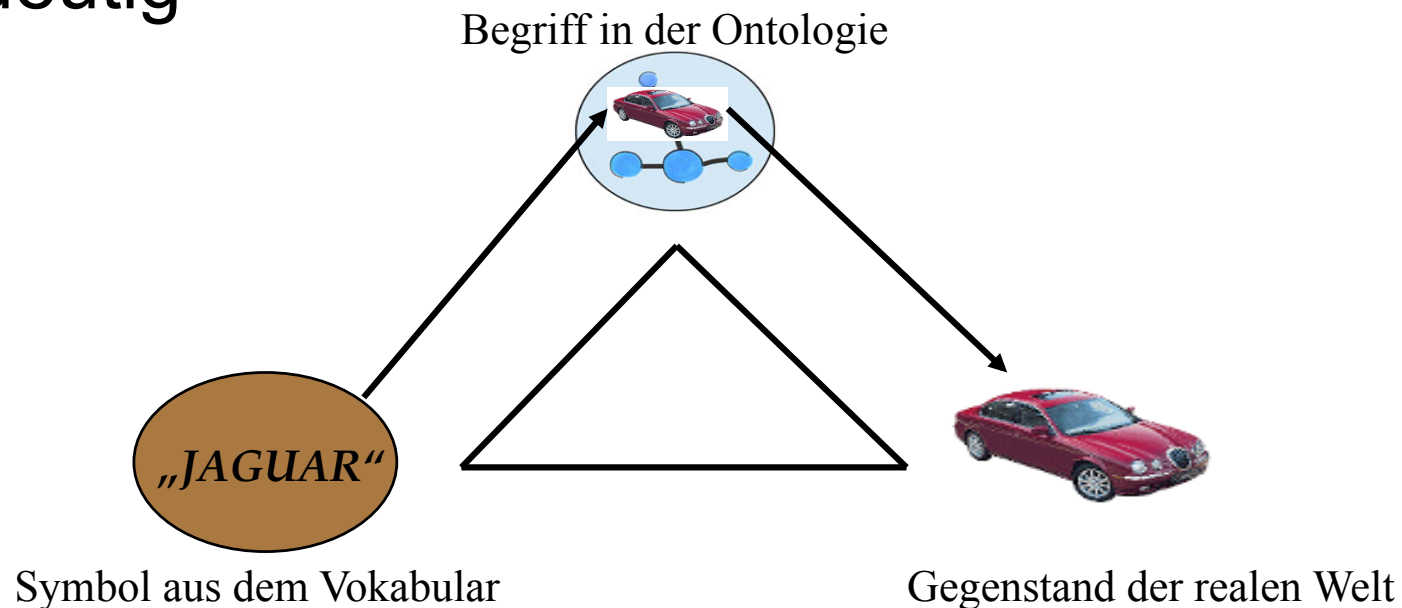
- Der allgemeine Zusammenhang wird von dem Bedeutungs-dreieck beschrieben, welches die Interaktion zwischen **Symbolen**, **Begriffen** und **Gegenständen** der Welt definiert.



Symbole und Gegenstände



- Die Ontologie reduziert die Anzahl von Abbildungen von Symbolen auf Gegenstände der realen Welt, im Idealfall ist die Abbildung eindeutig



Ontologiesprachen

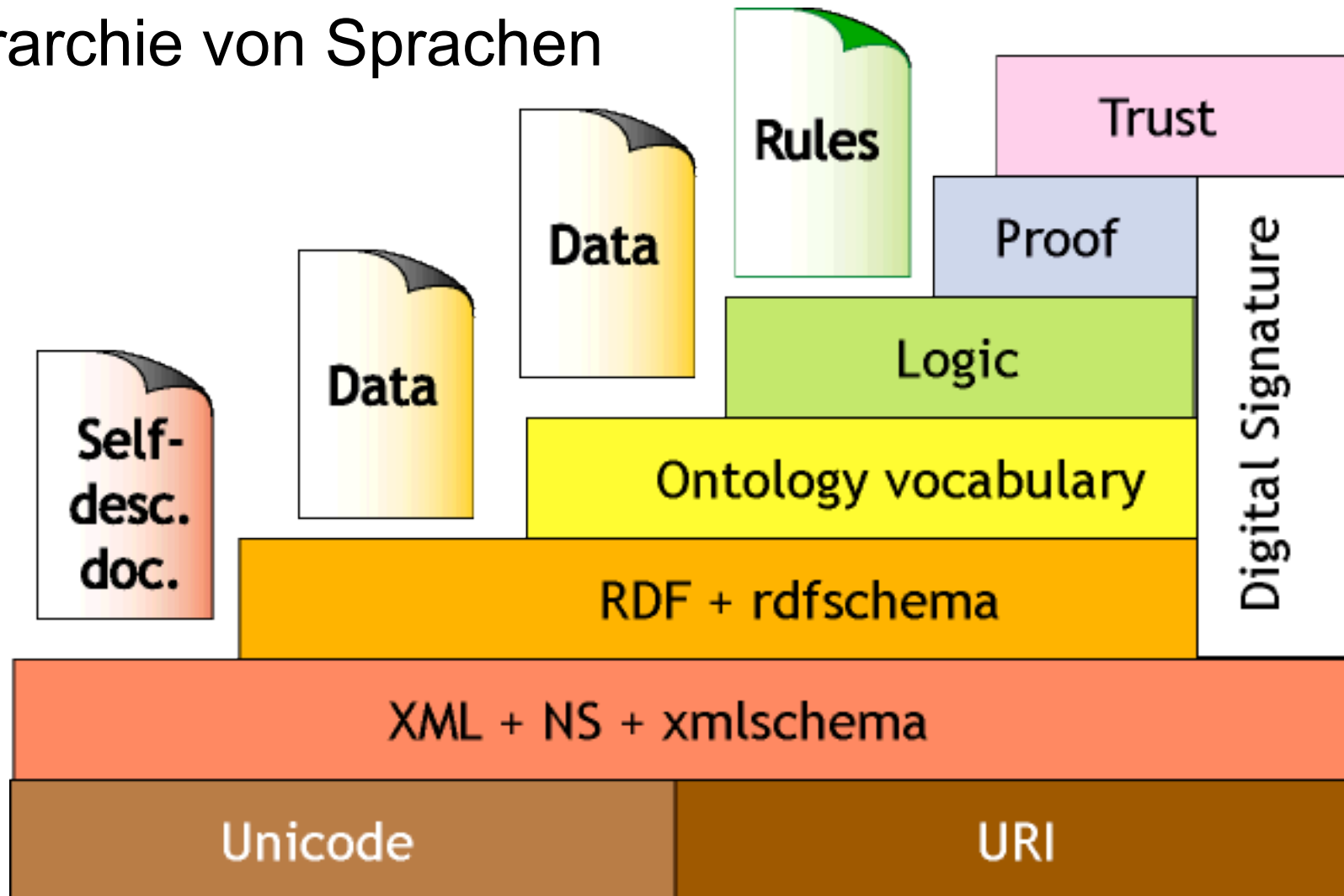
- Sprachen, mit denen man konzeptuell modellieren kann:
 - ⇒ Prädikatenlogik erster Stufe
 - ⇒ Frame-Logic
 - ⇒ RDF(S) und OWL (W3C Recommendations)
- Sprachen unterscheiden sich in ihrer Ausdrucksmächtigkeit
- Anforderung: Formale Semantik und Ausführbarkeit
- Ontologiesprachen sollten für das Semantic Web geeignet sein, siehe z.B. OWL

Semantic Web

- Das WWW ist zur Nutzung durch den Menschen bestimmt
- WWW basiert auf der Markupsprache HTML, die beschreibt:
 - ⇒ wie Informationen **dargestellt** werden sollen,
 - ⇒ wie Informationen miteinander **verknüpft** werden können,
 - ⇒ aber nicht, was diese Informationen **bedeuten**....
- Semantik Web ist das Web der nächste Generation, das zur Nutzung durch **den Menschen und Maschinen** bestimmt ist
 - ⇒ Die Herausforderung für die SW Sprache:
die Bedeutung von Informationen (Web Seiten, Web Links) in maschinelesbarer Form darzustellen

Semantic Web

Hierarchie von Sprachen



Beschreibungslogiken (DLs)

Beschreibungslogiken sind Formalismen zur Wissensrepräsentation

- Logikbasiert
 - ⇒ logikbasierte Semantik
 - ⇒ automatische Deduktion
- Ausdrucksstark für komplexes Wissen
- Schlank genug für Anwendbarkeit

- Grundlage für Ontologiesprache OWL (Web Ontology Language, W3C Standard April 2004)
- Kerntechnologie für das Semantic Web
- Grundlage für Semantisches Wissensmanagement in Unternehmen

Einführung in DLs

- Beschreibungslogiken (Description Logics, kurz DLs) sind eine **Familie von Formalismen** für explizite und implizite **Repräsentation** von **strukturiertem Wissen**.
- Beschreibungslogiken **vereinheitlichen** und **reichern** u.a. die traditionellen framebasierten, netzwerkartigen und objektorientierten Modellierungssprachen **mit formaler Semantik an**.

Fokus und Zielsetzung

- Je **ausdrucksmächtiger** die Logik **desto schwieriger** wird das automatische Schließen.
 - ⇒ Komplexität steigt
 - ⇒ Manche Probleme sind sogar unentscheidbar
- **Ziel** ist ein **gesundes Gleichgewicht** zu finden, d.h. eine möglichst **ausdrucksmächtige** Logik, die für wichtige Probleme **entscheidbares** und möglichst **effizientes** automatisches Schließen anbietet.

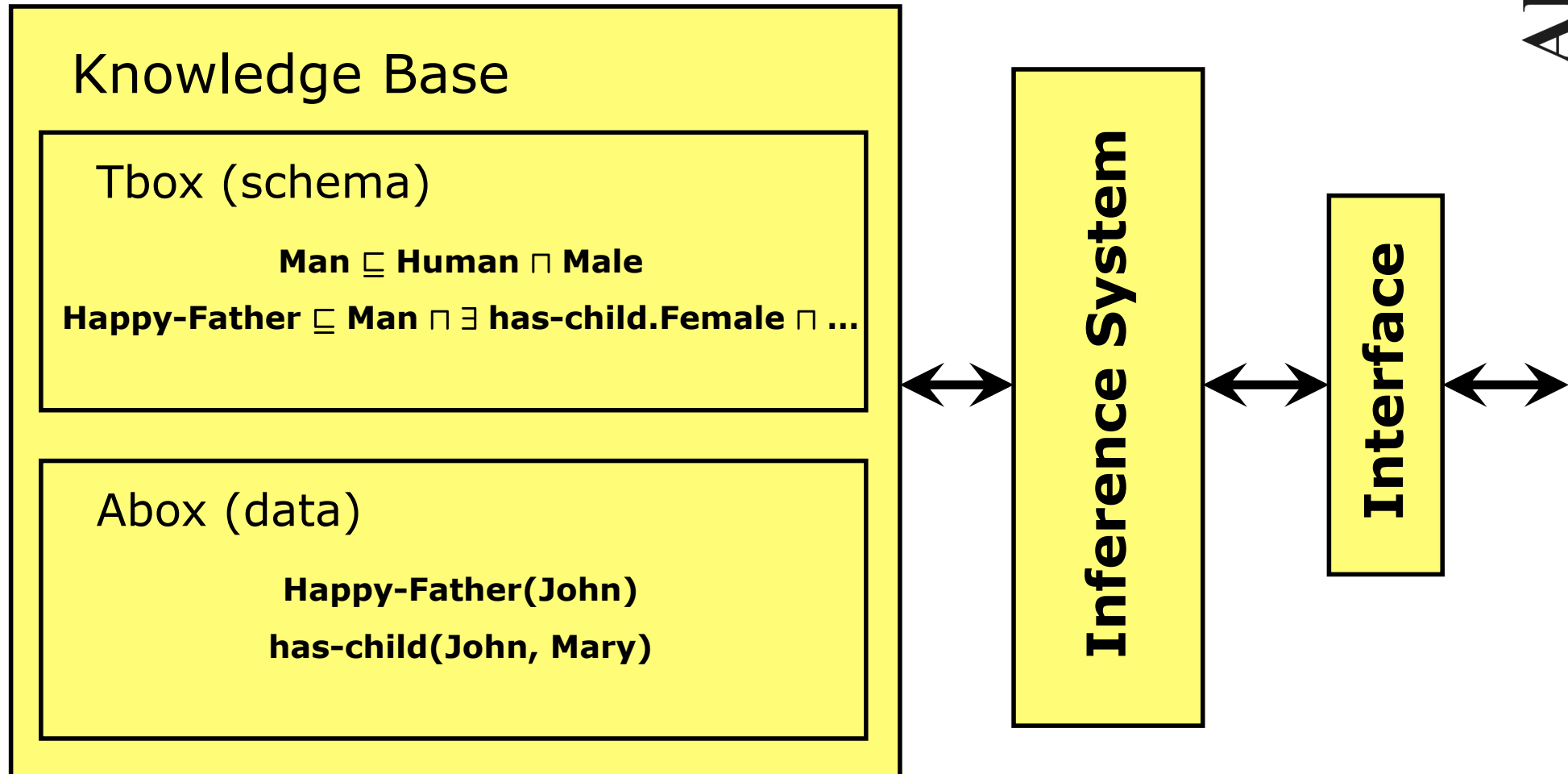
DL Syntax und Semantik

- Beschreibungslogiken sind eine Familie von Logiken.
 - ⇒ Es gibt nicht *die* Beschreibungslogik, sondern **viele verschiedene Beschreibungslogiken**.
- In DL werden mittels sog. **Konstruktoren** aus einfachen Beschreibungen komplexere Beschreibungen gebaut.
- Verschiedene Beschreibungslogiken **unterscheiden sich in der Menge der Konstrukturen (Ausdrucksmächtigkeit)**, die sie anbieten.

Konstrukturen

- Konstrukturen ermöglichen den **Aufbau** von **komplexeren Konzepten aus** weniger komplexeren bzw. **atomaren Konzepten**.
- Welche Arten von Konstrukturen es gibt, hängt von der konkreten DL ab.
- Die meisten DLs bieten jedoch
 - ⇒ Konjunktion (\sqcap), Disjunktion (\sqcup), Negation (\neg)
 - ⇒ Existenzquantor (\exists) und Allquantor (\forall)

Allgemeine DL Architektur



Einfaches Beispiel

Terminologisches Wissen (*TBox*):

Axiome, die die Struktur der zu modellierenden Domäne beschreiben (konzeptionelles Schema)

$\text{Human} \sqsubseteq \exists \text{hasParent.Human}$

$\text{Orphan} \equiv \text{Human} \sqcap \neg \exists \text{hasParent.Alive}$

Wissen um Individuen (*ABox*):

Axiome, die konkrete Situationen (Daten) beschreiben

$\text{Orphan}(\text{harrypotter})$

$\text{hasParent}(\text{harrypotter}, \text{jamespotter})$

ALC Atomare Typen

➤ Konzeptname / Klassenname

- ⇒ Beispiele: Student, Mitarbeiter, Professor, Vorlesung, Studiengang
- ⇒ vergleichbar mit einer Klasse in UML und einem Entitätstyp in ER.

➤ Rollenname

- ⇒ zweistellige Prädikate
(verbinden zwei Klassen oder Individuen)
- ⇒ Beispiele: Student *besucht* Vorlesung, Mitarbeiter *betreut* Vorlesung, Professor *hält* Vorlesung
- ⇒ vergleichbar mit Assoziation in UML und Beziehungstyp in ER

ABox Syntax

➤ Konzept Assertion $C(a)$

- ⇒ Beispiel: Student(Peter), Vorlesung(AngInfol)
- ⇒ Vergleichbar mit Objekten in UML und Entitäten in ER.

➤ Rolle Assertion $R(a, b)$

- ⇒ Beispiel: besucht(Peter, AngInfol)
- ⇒ Vergleichbar mit Assoziationen in UML und Beziehungen in ER.

- Eine **ABox** ist eine endliche Menge von solchen Axiomen (Konzept Assertion und Rolle Assertion).
- Die in ABox Axiomen benutzten Konzepte **können aber müssen nicht** in TBox definiert sein.

ALC TBox Syntax

- *ALC* ist die **einfachste** Beschreibungslogik
- Atomare Typen
 - ⇒ Konzept- (oder Klassen-) name A, B, \dots und folgende zwei spezielle Konzepte
 - ◇ \perp Bottom Konzept
 - ◇ \top Top oder universelles Konzept
 - ⇒ Rollennamen R, S, \dots
- Konstruktoren
 - ⇒ $\neg C$ (Negation)
 - ⇒ $C \sqcap D$ (Konjunktion)
 - ⇒ $C \sqcup D$ (Disjunktion)
 - ⇒ $\exists R.C$ (Existenzquantor)
 - ⇒ $\forall R.C$ (Allquantor)

ALC Klassenbeziehungen

- Klassen können auf zwei Arten miteinander in Beziehung gesetzt werden:

- ⇒ Inklusion

$$C \sqsubseteq D$$

- ✧ z.B. $\text{Man} \sqsubseteq \text{Human}$

- ⇒ Gleichheit

$$C \equiv D$$

- ✧ z.B. $\text{Frau} \equiv \text{Woman}$

- ✧ $C \equiv D$ ist gleichbedeutend damit, dass *beide* Aussagen $C \sqsubseteq D$ und $D \sqsubseteq C$ gelten

ALC **Konstrukturen**

- Negation von C bedeutet intuitiv “alles ausser C ”
 - ⇒ $\text{Mann} \equiv \neg \text{Frau}$
- Konjunktion von C und D bedeutet intuitiv “sowohl C als auch D ”
 - ⇒ $\text{Touchscreen} \equiv \text{Eingabegerät} \sqcap \text{Ausgabegerät}$
- Disjunktion von C und D bedeutet intuitiv “ C oder D ”
 - ⇒ $\text{UniAngestellte} \equiv \text{Mitarbeiter} \sqcup \text{Professor}$

ALC **Konstrukturen**

- $\exists R.C$ (Existenzquantor) bedeutet intuitiv „alle x , die eine Beziehung vom Typ R mit einem y vom Typ C haben“

⇒ \exists besucht.Vorlesung

- $\forall R.C$ (Allquantor) bedeutet intuitiv „alle x , für die alle Beziehungen vom Typ R mit einem y vom Typ C sind“

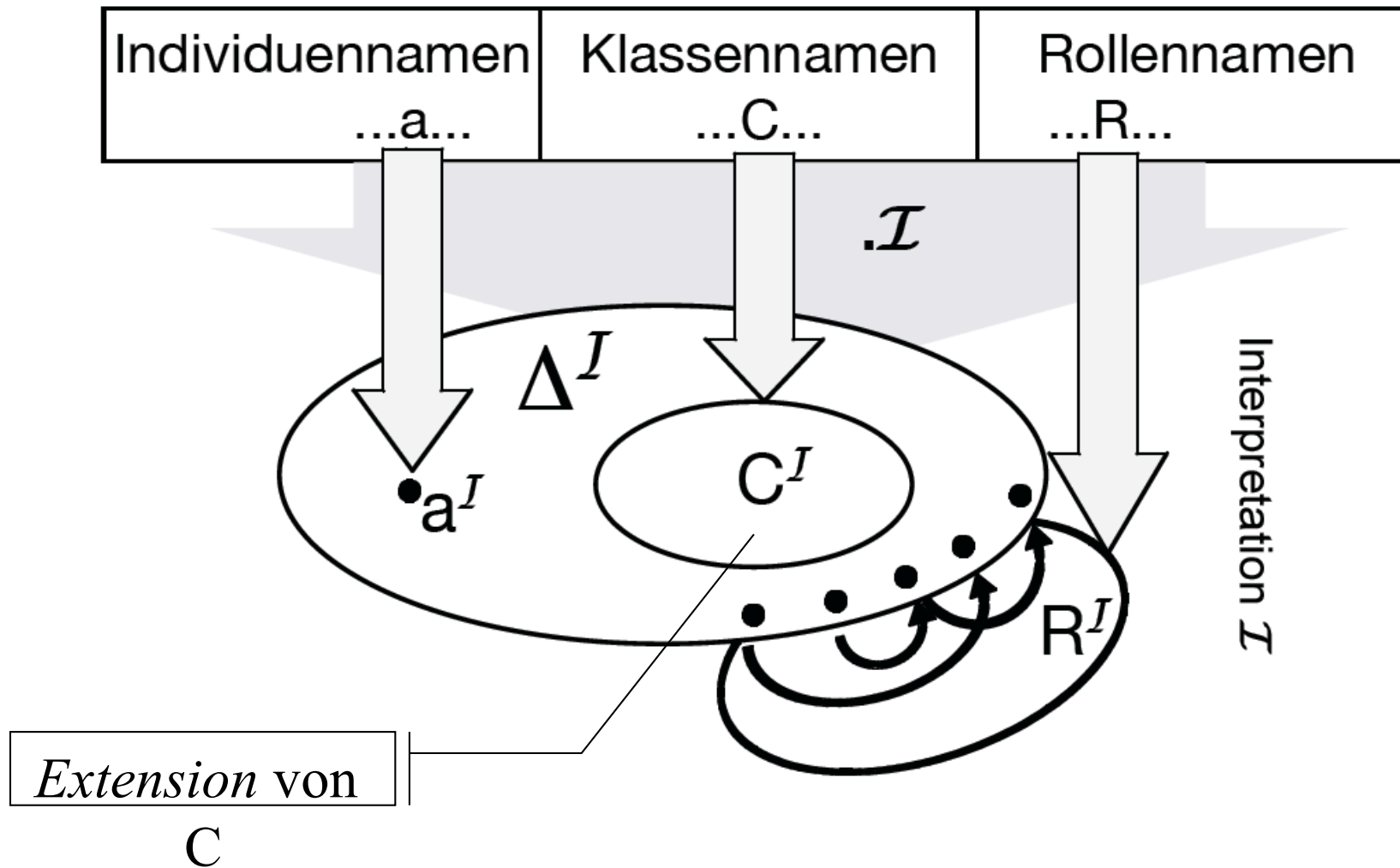
⇒ \forall hat Elternteil.Mensch

⇒ hat ein Individuum gar keine Beziehung vom Typ R , erfüllt es immer die Bedingung $\forall R.C$.

ALC Klassenbeziehungen

- Klassenbeziehungen können für zusammengesetzte Klassen verwendet werden:
 - ⇒ Inklusion
 - $C \sqsubseteq D$
 - ✧ z.B. $\text{Man} \sqsubseteq \text{Human} \sqcap \text{Male}$
 - ⇒ Gleichheit
 - $C \equiv D$
 - ✧ z.B. $\text{Frau} \equiv \text{Human} \sqcap \text{Female} \sqcap \text{Adult}$
 - ✧ z.B. $\text{Orphan} \equiv \text{Human} \sqcap \forall \text{hasParent.Dead}$
- Terminologisches Wissen (sog. **TBox**) besteht aus einer Menge solcher Klassenbeziehungen (sog. *Axiome*).

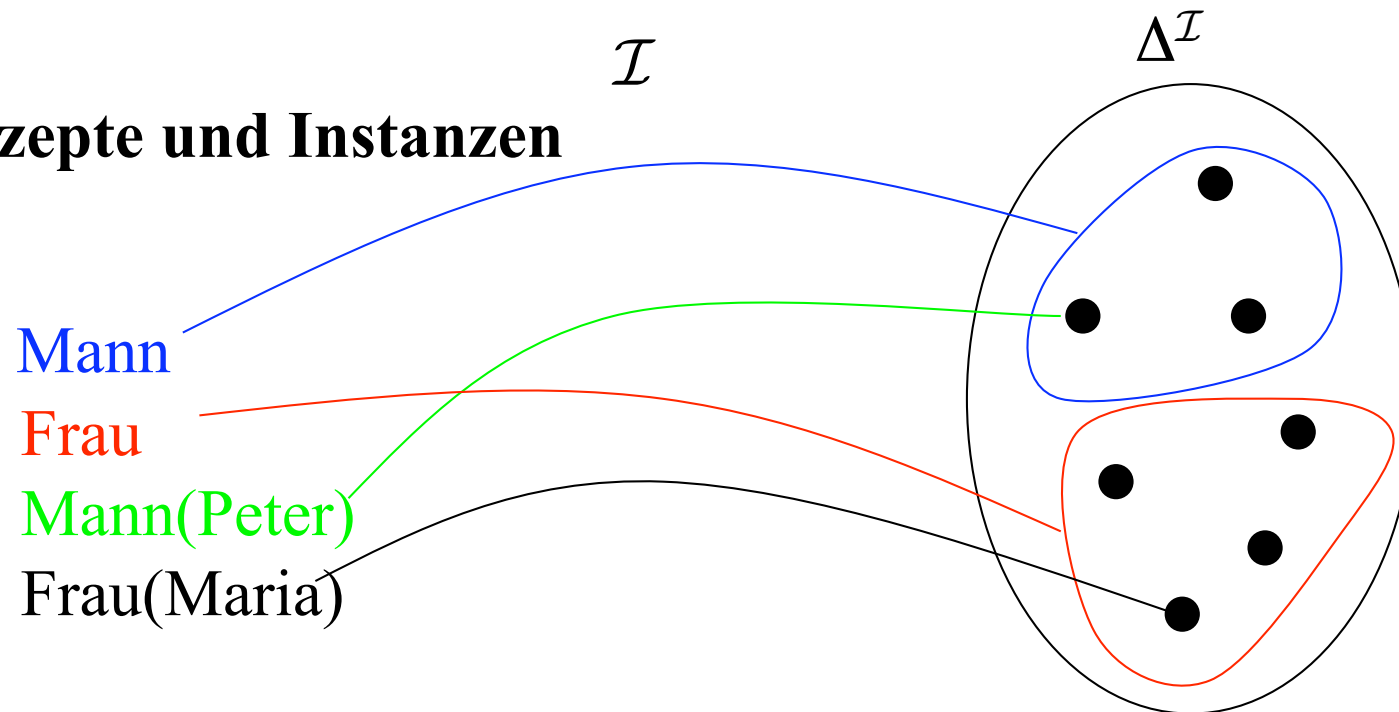
Interpretationen für \mathcal{ALC}



- Eine Interpretation \mathcal{I} erfüllt

⇒ $C(a)$ gdw. $a^{\mathcal{I}} \in C^{\mathcal{I}}$

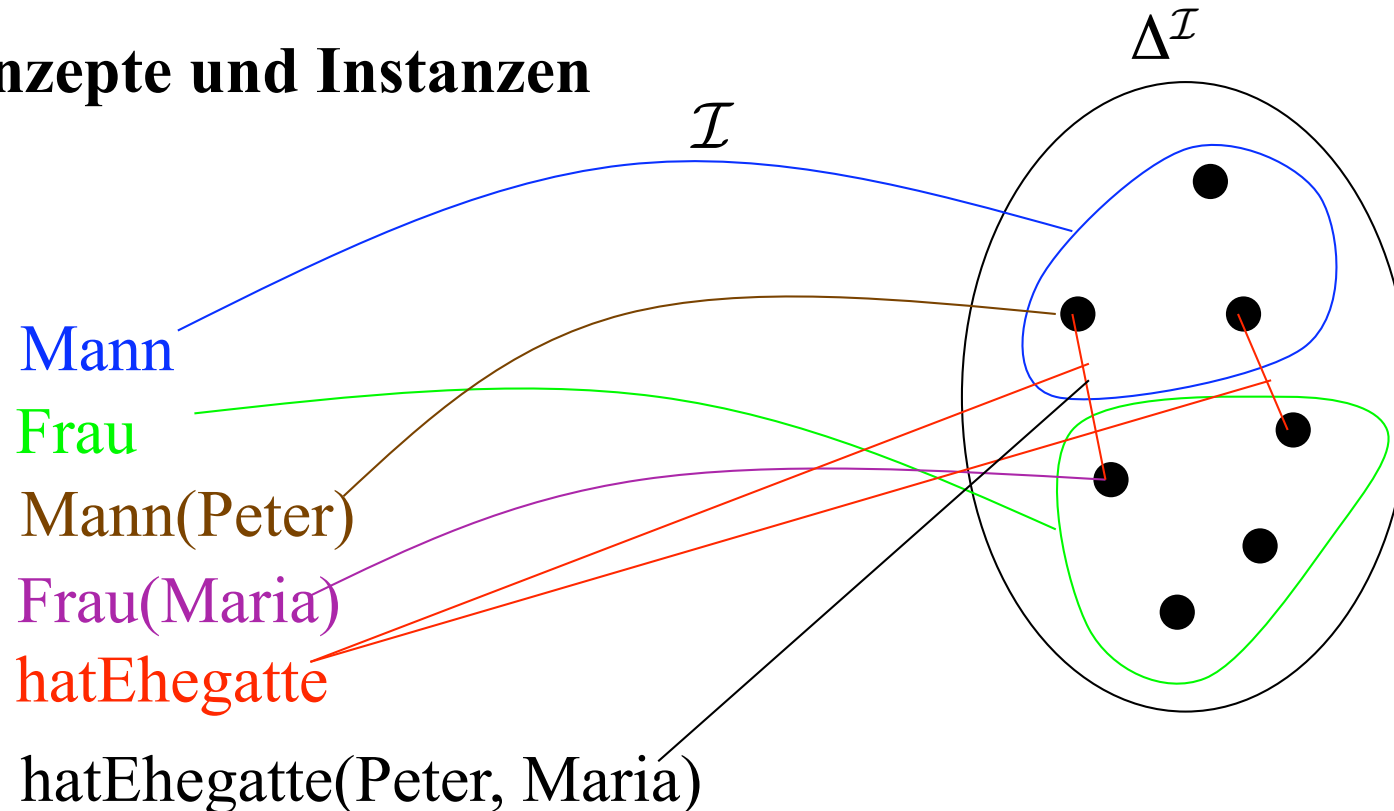
Konzepte und Instanzen



ABox Semantik

- Eine Interpretation \mathcal{I} erfüllt
 - ⇒ $R(a, b)$ gdw. $(a^{\mathcal{I}}, b^{\mathcal{I}}) \in R^{\mathcal{I}}$

Konzepte und Instanzen



\mathcal{ALC} TBox Semantik

Klassenbeziehungen:

➤ Gleichheit: $A \equiv C$

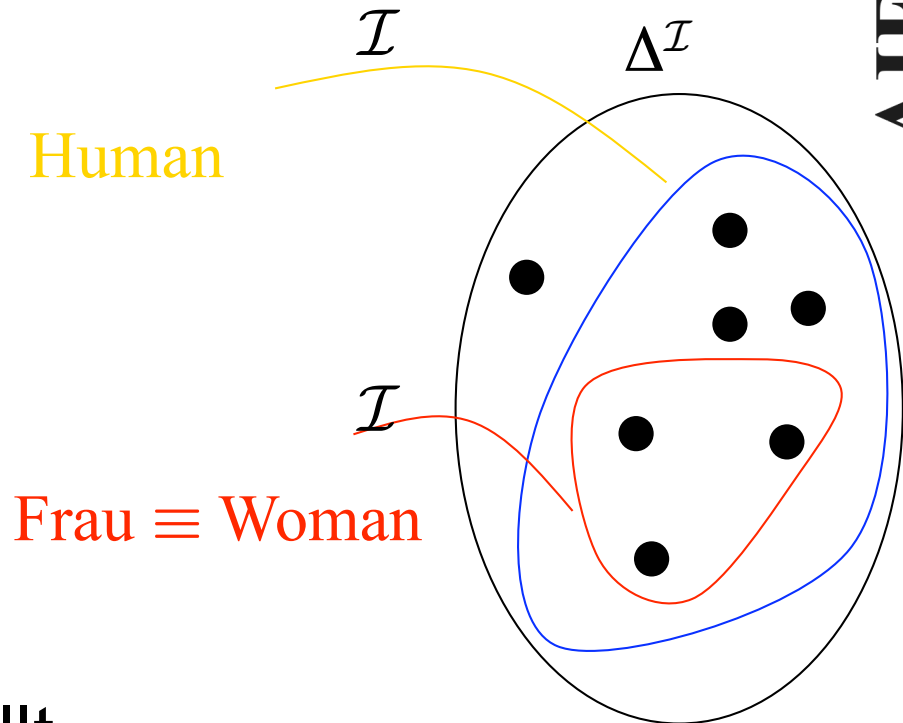
➤ Inklusion: $C \sqsubseteq D$

➤ Eine Interpretation \mathcal{I} erfüllt

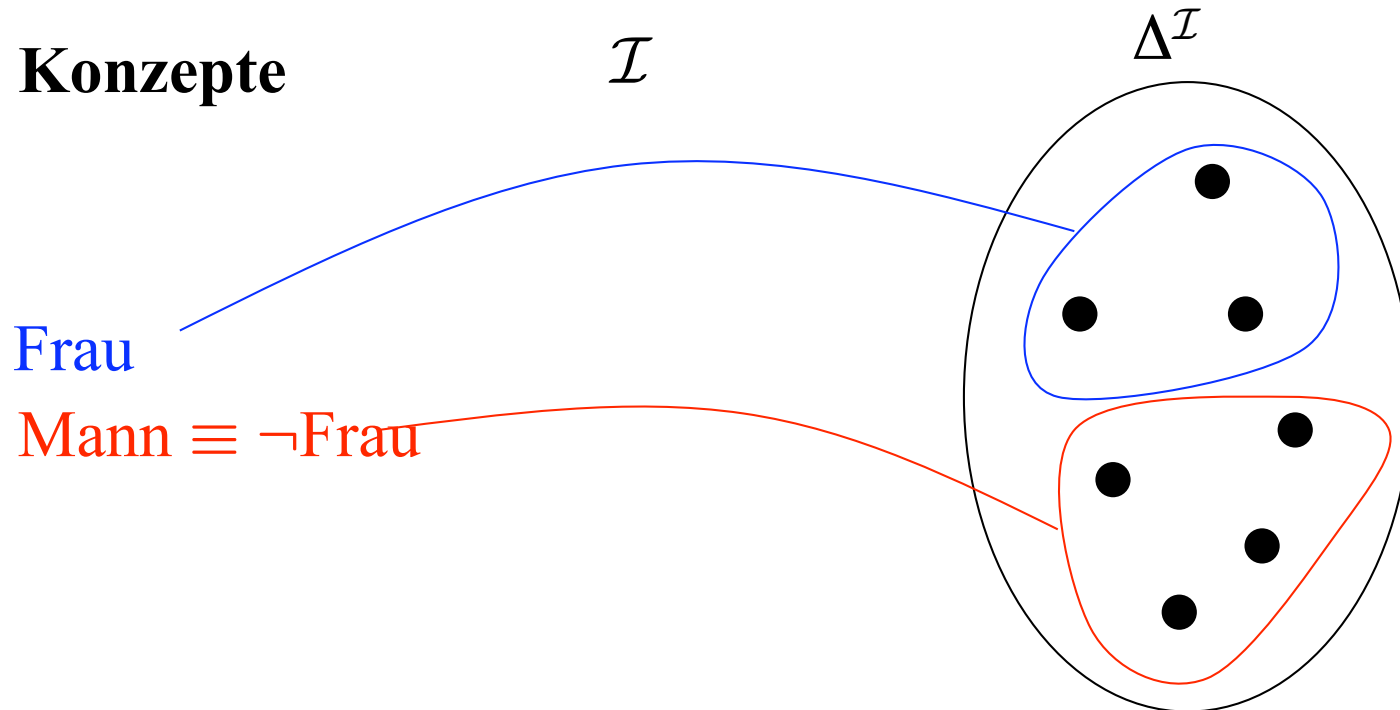
⇒ $C \equiv D$ gdw. $C^{\mathcal{I}} = D^{\mathcal{I}}$

⇒ $C \sqsubseteq D$ gdw. $C^{\mathcal{I}} \subseteq D^{\mathcal{I}}$

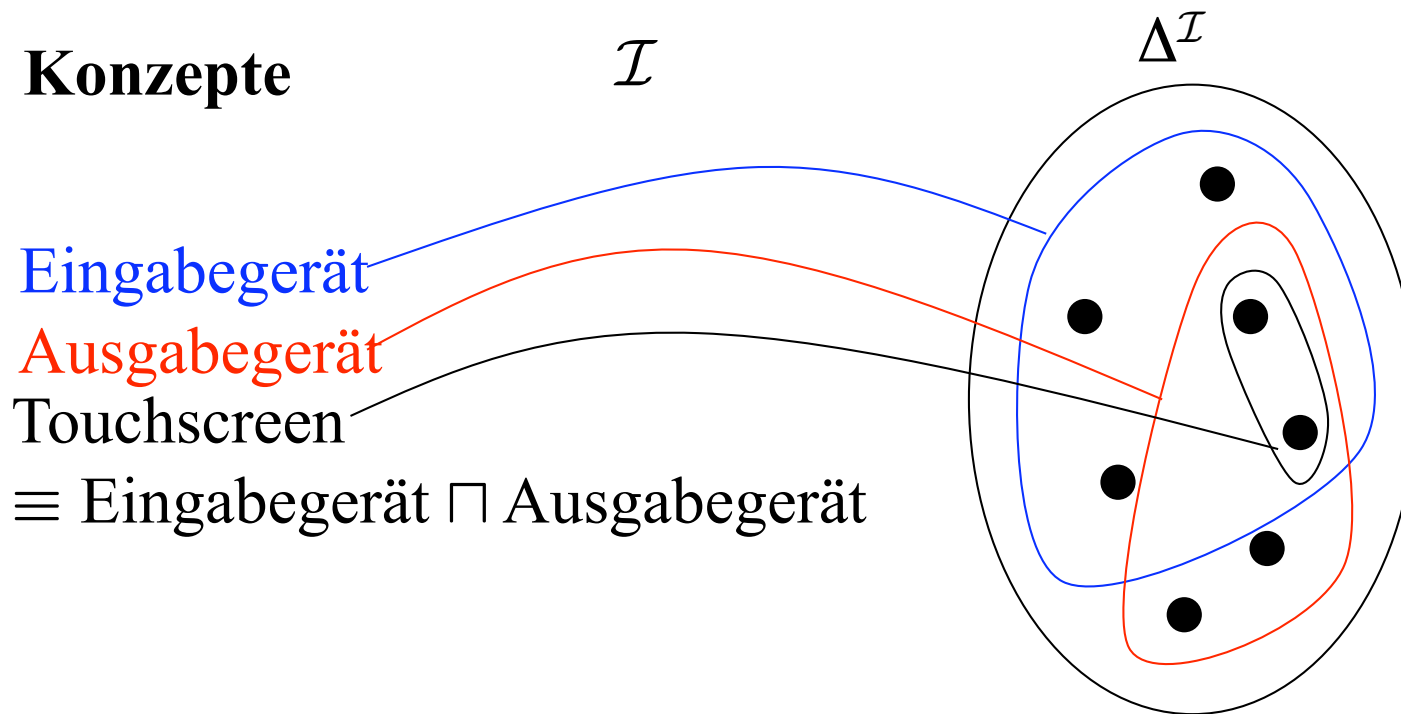
⇒ eine TBox \mathcal{T} gdw. \mathcal{I} jedes Axiom in \mathcal{T} erfüllt



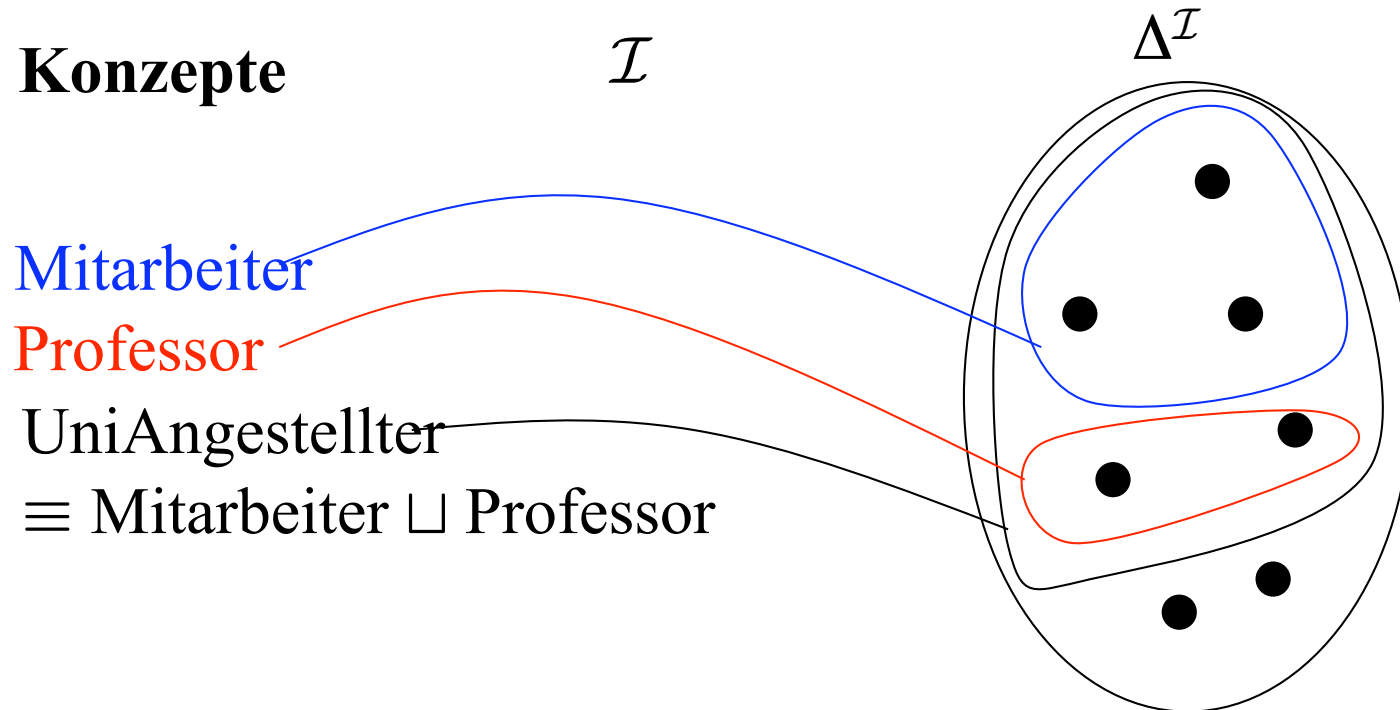
- Semantik der Negation $\neg(C)^{\mathcal{I}} = \Delta^{\mathcal{I}} \setminus C^{\mathcal{I}}$



- Semantik der Konjunktion $(C \sqcap D)^{\mathcal{I}} = C^{\mathcal{I}} \cap D^{\mathcal{I}}$



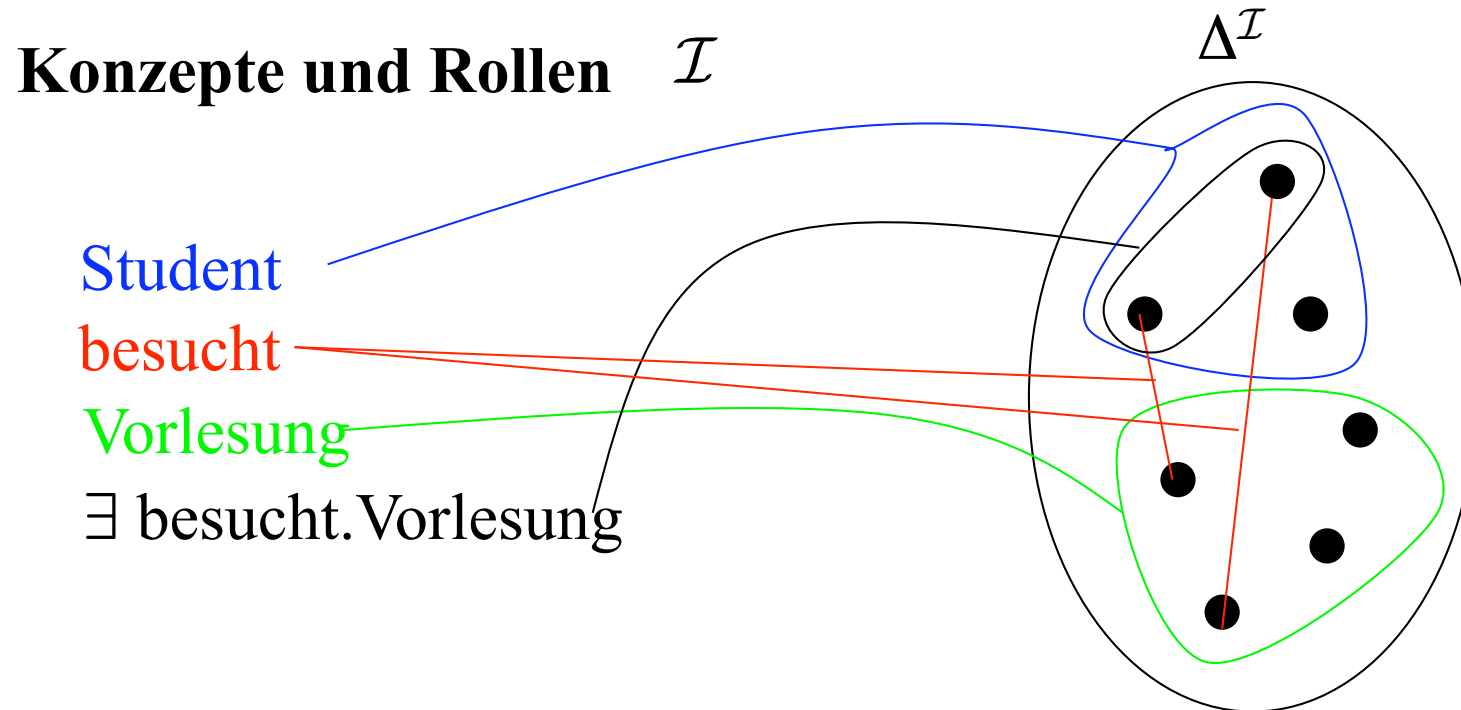
- Semantik der Disjunktion $(C \sqcup D)^{\mathcal{I}} = C^{\mathcal{I}} \cup D^{\mathcal{I}}$



\mathcal{ALC} TBox Semantik

➤ Semantik des Existenzquantors

$$\Rightarrow (\exists R.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \exists e \in \Delta^{\mathcal{I}} \text{ mit } (d,e) \in R^{\mathcal{I}} \wedge e \in C^{\mathcal{I}}\}$$

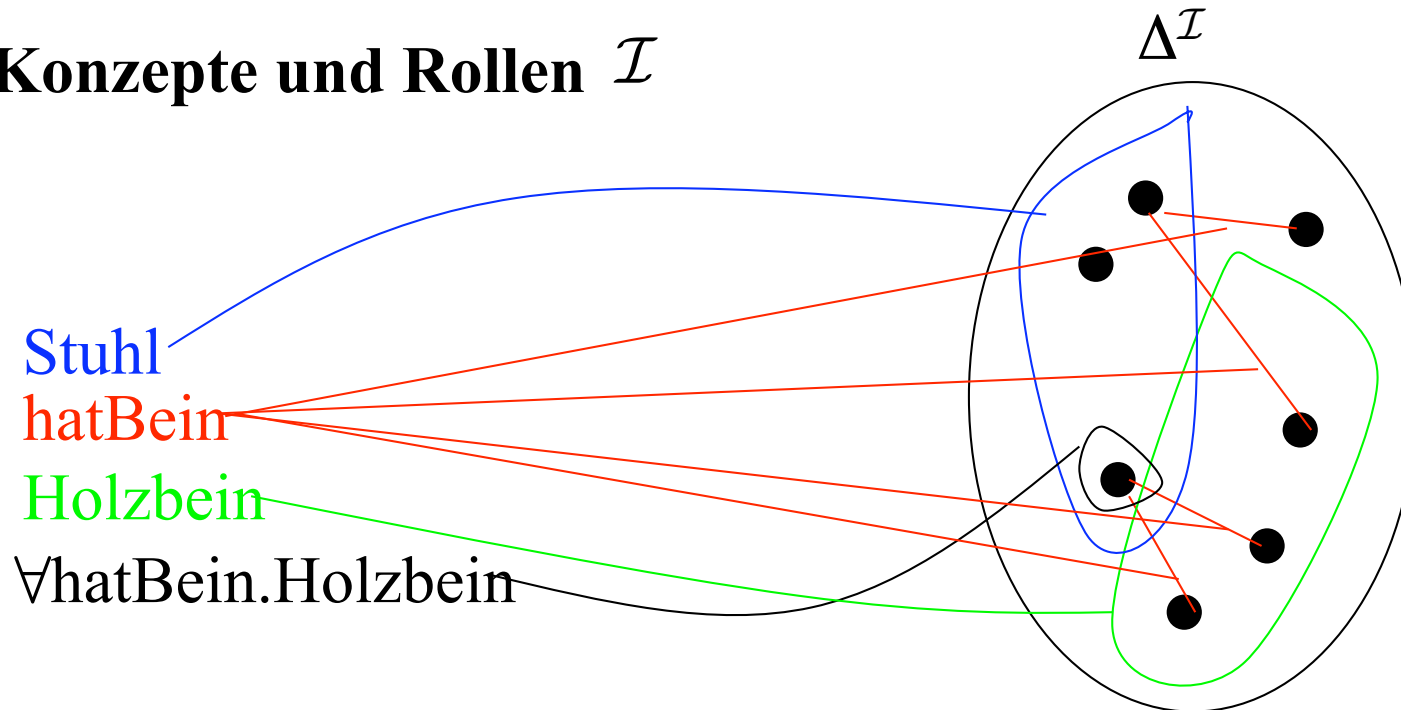


\mathcal{ALC} TBox Semantik

➤ Semantik des Allquantors

$$\Rightarrow (\forall R.C)^{\mathcal{I}} = \{d \in \Delta^{\mathcal{I}} \mid \forall e \in \Delta^{\mathcal{I}} (d,e) \in R^{\mathcal{I}} \Rightarrow e \in C^{\mathcal{I}}\}$$

Konzepte und Rollen \mathcal{I}



Erweiterung von \mathcal{ALC}

Mit der Zeit sind verschiedene DL Konstruktoren eingeführt worden, um \mathcal{ALC} zu erweitern. Z.B.

➤ Zahlenrestriktionen:

⇒ Syntax:

◇ $\geq n R$

◇ $\leq n R$

⇒ Semantik:

◇ $\{x \mid |\{\langle x, y \rangle \in R^I\}| \geq n\}$

◇ $\{x \mid |\{\langle x, y \rangle \in R^I\}| \leq n\}$

⇒ Beispiele:

◇ ≥ 3 hatKind bedeutet alle Individuen, die mindestens drei Kinder haben

◇ ≤ 1 hatMutter bedeutet alle Individuen, die höchstens eine Mutter haben.

Erweiterung von \mathcal{ALC}

- qualifizierte Zahlenrestriktionen: wie Zahlenrestriktion nur mit dem Unterschied, dass der Wertebereich festgelegt ist
 - ⇒ Syntax: $\geq n \text{ R.C}$ bzw. $\leq n \text{ R.C}$
 - ⇒ Semantik: $\{x \mid |\{\langle x, y \rangle \in R^I \wedge y \in C^I\}| \geq n\}$ bzw. $\{x \mid |\{\langle x, y \rangle \in R^I \wedge y \in C^I\}| \leq n\}$
 - ⇒ Beispiele:
 - ✧ $\geq 2 \text{ hatKind.Frau}$ bedeutet alle Individuen, die mindestens 2 weibliche Kinder (Töchter) haben.
 - ✧ $\leq 1 \text{ hatElternteil.Mann}$ bedeutet alle Individuen, die höchstens einen männlichen Elternteil (Vater) haben.

Erweiterung von \mathcal{ALC}

➤ Subrollenbeziehung

⇒ Syntax: $R \sqsubseteq S$

⇒ Semantik: $R^{\mathcal{I}} \subseteq S^{\mathcal{I}}$

⇒ Beispiel: $\text{istVaterVon} \sqsubseteq \text{hatKind}$ bedeutet, dass wann immer a Vater von b ist, a auch b als Kind hat.

➤ Rollenäquivalenz

⇒ Syntax: $R \equiv S$

⇒ Semantik: $R^{\mathcal{I}} = S^{\mathcal{I}}$

⇒ Beispiel: $\text{istVorfahrVon} \equiv \text{hatNachfahren}$ bedeutet, dass a Vorfahr von b genau dann ist, wenn a b als Nachfahren hat.

Erweiterung von \mathcal{ALC}

➤ inverse Rollen

⇒ Syntax: R^-

⇒ Semantik: $\{\langle x, y \rangle \mid \langle y, x \rangle \in R^{\mathcal{I}}\}$

⇒ Beispiel: $\text{hatKind}^- \equiv \text{hatElternteil}$ bedeutet, dass hatElternteil eine inverse Rolle von hatKind ist.
D.h. wenn a Kind von b ist, ist b Elternteil von a .

➤ transitive Rollen

⇒ Syntax : $\text{Trans}(R)$

⇒ Semantik: $R^{\mathcal{I}} = (R^{\mathcal{I}})^*$

⇒ Beispiel: $\text{Trans}(\text{hatNachkommen})$ bedeutet, dass die Nachkommen von Nachkommen auch Nachkommen sind. D.h. wenn b Nachkommen von a ist und c Nachkommen von b , dann ist c Nachkommen von a .

Erweiterung von \mathcal{ALC}

- Nominal: Damit kann man vorgeben welche Individuen Instanz von einem Konzept sein dürfen.
 - ⇒ Syntax: $\{a_1, \dots, a_n\}$
 - ⇒ Semantik: $\{a_1, \dots, a_n\}^{\mathcal{I}} = \{a_1^{\mathcal{I}}, \dots, a_n^{\mathcal{I}}\}$
 - ⇒ Beispiel: EU-Mitglied $\equiv \{\text{Deutschland, Italien, ...}\}$

- Konkrete Rollen: Rollen, die als zweite Argumente Datentypen wie z.B. Zahlen oder Strings haben.
 - ⇒ z.B. hasAge(Markus, 25)
 - ⇒ dabei ist die Semantik der Datentypen festgelegt, d.h. $25^{\mathcal{I}}$ ist die Zahl 25.

Erweiterungen von \mathcal{ALC}

Concepts		
\mathcal{ALC}	Atomic	A, B
	Not	$\neg C$
	And	$C \sqcap D$
	Or	$C \sqcup D$
	Exists	$\exists R.C$
	For all	$\forall R.C$
$\mathcal{Q}(\mathcal{N})$	At least	$\geq n \ R.C \ (\geq n \ R)$
	At most	$\leq n \ R.C \ (\leq n \ R)$
\mathcal{O}	Nominal	$\{i_1, \dots, i_n\}$

Roles		
\mathcal{I}	Atomic	R
	Inverse	R^-

Concept Axioms (TBox)	
Subclass	$C \sqsubseteq D$
Equivalent	$C \equiv D$

Role Axioms (RBox)	
\mathcal{H} Subrole	$R \sqsubseteq S$
\mathcal{S} Transitivity	$\text{Trans}(S)$

Assertional Axioms (ABox)	
Instance	$C(a)$
Role	$R(a, b)$
Same	$a = b$
Different	$a \neq b$

$\mathcal{S} = \mathcal{ALC} + \text{Transitivity}$ $\text{OWL DL} = \mathcal{SHOIN}(\mathcal{D})$ (\mathcal{D} : Datentypen)

Wissensmodellierung (Bsp.)

cow \sqsubseteq vegetarian

vegetarian $\equiv \forall \text{eats}. \neg \exists \text{partof. animal} \sqcap \forall \text{eats}. \neg \text{animal}$

madcow $\equiv \text{cow} \sqcap \exists \text{eats}. (\exists \text{partof. sheep} \sqcap \text{brain})$

sheep \sqsubseteq animal

madcow(benita)

- Cows are vegetarians.
- Vegetarians eat neither animal parts nor animals.
- A mad cow is one that has been eating sheep brains.
- Sheep are animals.
- Benita is a mad cow.

Schlussfolgerung: Wissensbasis ist inkonsistent!

DL und FOL

(FOL = Prädikatenlogik 1. Stufe)

\mathcal{ALC} (ebenso andere DLs) ist ein Fragment von FOL

- jede Wissensbasis kann semantikerhaltend nach FOL übersetzt werden
- Instrumentarium der Logik (aus Mathematik, Philosophie, Informatik, Künstlicher Intelligenz) kann verwendet werden.
(über 2000 Jahre Entwicklung seit Aristoteles)

➤ Umformungen nach FOL

$$C \equiv D \quad \Leftrightarrow \quad (\forall x) (C(x) \leftrightarrow D(x))$$

$$C \sqsubseteq D \quad \Leftrightarrow \quad (\forall x) (C(x) \rightarrow D(x))$$

$$C \sqcap D \quad \Leftrightarrow \quad C(x) \wedge D(x)$$

$$C \sqcup D \quad \Leftrightarrow \quad C(x) \vee D(x)$$

$$\neg C \quad \Leftrightarrow \quad \neg C(x)$$

$$\forall R.C \quad \Leftrightarrow \quad (\forall y) (R(x,y) \rightarrow C(y))$$

$$\exists R.C \quad \Leftrightarrow \quad (\exists y) (R(x,y) \wedge C(y))$$

Umformung nach FOL

cow \sqsubseteq vegetarian

madcow \equiv cow \sqcap \exists eats. (\exists partof. sheep \sqcap brain)

wird zu

$(\forall x) (\text{cow}(x) \rightarrow \text{vegetarian}(x))$

$(\forall x) (\text{madcow}(x) \leftrightarrow$

$(\text{cow}(x) \wedge (\exists y)(\text{eats}(x,y) \wedge$

$(\exists z)(\text{partof}(y,z) \wedge (\text{sheep}(z) \wedge \text{brain}(z)))$

$)$

$)$

$)$

Wichtige Inferenzprobleme

- Globale Konsistenz der Wissensbasis KB \models **false**?
 - ⇒ Ist Wissensbasis sinnvoll?
- Klassenkonsistenz C $\equiv \perp$?
 - ⇒ Muss Klasse C leer sein?
- Klasseninklusion (Subsumption) C \sqsubseteq D?
 - ⇒ Strukturierung der Wissensbasis
- Klassenäquivalenz C \equiv D?
 - ⇒ Sind zwei Klassen eigentlich dieselbe?
- Klassendisjunktheit C \sqcap D = \perp ?
 - ⇒ Sind zwei Klassen disjunkt?
- Klassenzugehörigkeit C(a)?
 - ⇒ Ist Individuum a in der Klasse C?
- Instanzgenerierung (Retrieval) „alle X mit C(X) finden“
 - ⇒ Finde alle (bekannten!) Individuen zur Klasse C.

Rückführung auf Unerfüllbarkeit

- **Klassenkonsistenz** $C \equiv \perp$
 - ⇒ gdw. $KB \cup \{C(a)\}$ unerfüllbar (a neu)
- **Klasseninklusion (Subsumption)** $C \sqsubseteq D$
 - ⇒ gdw. $KB \cup \{(C \sqcap \neg D)(a)\}$ unerfüllbar (a neu)
- **Klassenäquivalenz** $C \equiv D$
 - ⇒ gdw. $C \sqsubseteq D$ und $D \sqsubseteq C$
- **Klassendisjunktheit** $C \sqcap D \sqsubseteq \perp$
 - ⇒ gdw. $KB \cup \{(C \sqcap D)(a)\}$ unerfüllbar (a neu)
- **Klassenzugehörigkeit** $C(a)$
 - ⇒ gdw. $KB \cup \{\neg C(a)\}$ unerfüllbar (a neu)
- **Instanzgenerierung (Retrieval)** alle $C(X)$ finden
 - ⇒ prüfe Klassenzugehörigkeit für alle Individuen

Tableauverfahren

- Wir werden das Tableauverfahren behandeln
 - Das Tableauverfahren versucht, auf "abstrakte" Weise ein Modell zu konstruieren.
 - ⇒ misslingt das, dann ist die Wissensbasis unerfüllbar
 - ⇒ gelingt es, ist sie erfüllbar
- Rückführung der Inferenzprobleme auf das Zeigen der Unerfüllbarkeit der Wissensbasis!

Umformen in Negationsnormalform

Gegeben eine Wissensbasis W .

- Ersetze $C \equiv D$ durch $C \sqsubseteq D$ und $D \sqsubseteq C$.
- Ersetze $C \sqsubseteq D$ durch $\neg C \sqcup D$.
- Wende die Regeln auf der folgenden Folie an, bis es nicht mehr geht.

Resultierende Wissensbasis: $NNF(W)$

Negationsnormalform von W .

Negation steht nur noch direkt vor atomaren Klassen.

Umformen in Negationsnormalform

$$\begin{aligned}
 \text{NNF}(C) &= C, \text{ falls } C \text{ atomar ist} \\
 \text{NNF}(\neg C) &= \neg C, \text{ falls } C \text{ atomar ist} \\
 \text{NNF}(\neg\neg C) &= \text{NNF}(C) \\
 \text{NNF}(C \sqcup D) &= \text{NNF}(C) \sqcup \text{NNF}(D) \\
 \text{NNF}(C \sqcap D) &= \text{NNF}(C) \sqcap \text{NNF}(D) \\
 \text{NNF}(\neg(C \sqcup D)) &= \text{NNF}(\neg C) \sqcap \text{NNF}(\neg D) \\
 \text{NNF}(\neg(C \sqcap D)) &= \text{NNF}(\neg C) \sqcup \text{NNF}(\neg D) \\
 \text{NNF}(\forall R.C) &= \forall R.\text{NNF}(C) \\
 \text{NNF}(\exists R.C) &= \exists R.\text{NNF}(C) \\
 \text{NNF}(\neg\forall R.C) &= \exists R.\text{NNF}(\neg C) \\
 \text{NNF}(\neg\exists R.C) &= \forall R.\text{NNF}(\neg C)
 \end{aligned}$$

W und $\text{NNF}(W)$ sind logisch äquivalent.

Naives Tableauverfahren

Test auf Unerfüllbarkeit

Idee:

- Gegeben Wissensbasis W .
- Versuch der Erzeugung eines Baumes (genannt *Tableau*), der ein Modell von W repräsentiert.
(Eigentlich ein *Wald*.)
- Schlägt der Versuch fehl, ist W unerfüllbar.

Das Tableau

- Ein Tableau ist ein gerichteter beschrifteter Graph
- Knoten repräsentieren Elemente der Domäne $\Delta^{\mathcal{I}}$ des Modells
 - ⇒ Jeder Knoten x ist mit einer Menge $L(x)$ von Klassenausdrücken beschriftet.
 - ⇒ $C \in L(x)$ steht für: " x ist in der Extension von C "
- Kanten stehen für Rollenverbindungen
 - ⇒ Jede Kante $\langle x, y \rangle$ ist mit einer Menge von Rollennamen $L(\langle x, y \rangle)$ beschriftet.
 - ⇒ $R \in L(\langle x, y \rangle)$ steht für: " (x, y) ist in der Extension von R "

Initialisierung

- Input: $K = TBox + ABox$ (in NNF)
- Ein Knoten für jedes Individuum in der ABox.
- Jeder Knoten beschriftet mit den entsprechenden Klassennamen aus der ABox.
- Eine mit R beschriftete Kante zwischen a und b falls $R(a,b)$ in der ABox steht.

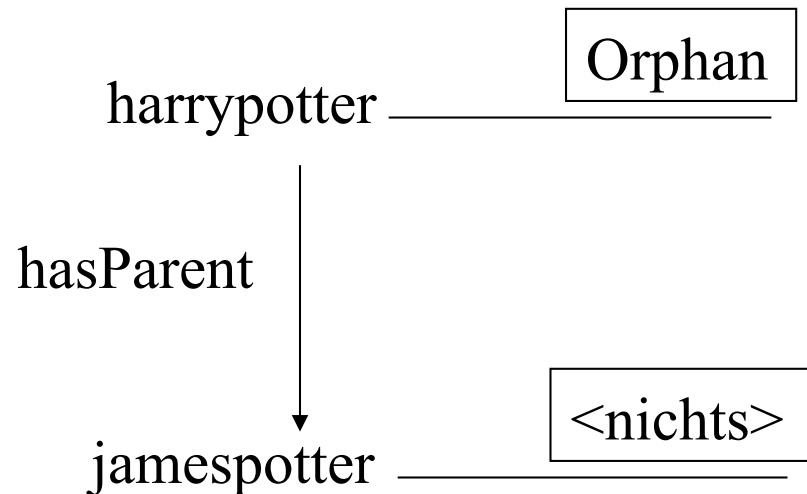
Beispiel: Initialisierung

Human $\sqsubseteq \exists \text{hasParent.Human}$

Orphan $\sqsubseteq \text{Human} \sqcap \neg \exists \text{hasParent.Alive}$

Orphan(harrypotter)

hasParent(harrypotter,jamespotter)



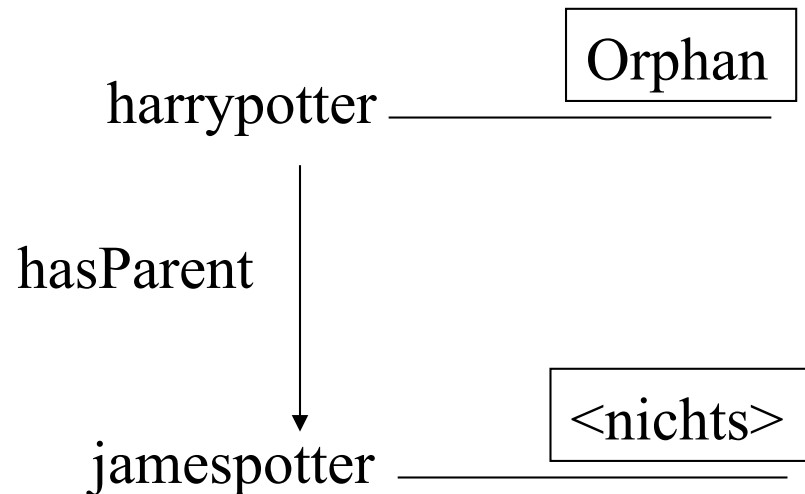
Negationsnormalform!

$\neg \text{Human} \sqcup \exists \text{hasParent. Human}$

$\neg \text{Orphan} \sqcup (\text{Human} \sqcap \forall \text{hasParent.} \neg \text{Alive})$

$\text{Orphan}(\text{harrypotter})$

$\text{hasParent}(\text{harrypotter}, \text{jamespotter})$



Konstruktion des Tableaus

- Erweitere das Tableau nichtdeterministisch mit den Regeln auf der folgenden Seite.
- Terminiere, wenn
 - ⇒ die Beschriftung eines Knotens einen Widerspruch enthält (d.h. Klassen C und $\neg C$ enthält) oder
 - ⇒ keine der Regeln mehr anwendbar ist
- Falls das Tableau keinen Widerspruch enthält, ist die Wissensbasis erfüllbar.
(genauer: sind die Regeln durch geeignete Auswahl so anwendbar, dass kein Widerspruch entsteht und keine Regel mehr anwendbar ist, so ist die Wissensbasis erfüllbar)

Tableauregeln

\sqcap : Wenn $C \sqcap D \in L(x)$ und $\{C, D\} \not\subseteq L(x)$, dann setze

$$L(x) = L(x) \cup \{C, D\}$$

\sqcup : Wenn $C \sqcup D \in L(x)$ und $\{C, D\} \cap L(x) = \emptyset$, dann setze

$$L(x) = L(x) \cup \{C\} \text{ oder } L(x) = L(x) \cup \{D\}$$

\exists : Wenn $\exists R. C \in L(x)$ und x keinen R -Nachfolger y mit $C \in L(y)$ hat, dann
 füge einen neuen Knoten y hinzu mit

$$L(\langle x, y \rangle) = \{R\} \text{ und } L(y) = C$$

\forall : Wenn $\forall R. C \in L(x)$ und es gibt einen R -Nachfolger y von x mit $C \notin L(y)$, dann setze

$$L(y) = L(y) \cup \{C\}$$

TBox: Ist C in der TBox und $C \notin L(x)$, dann setze

$$L(x) = L(x) \cup \{C\}$$

Beispiel

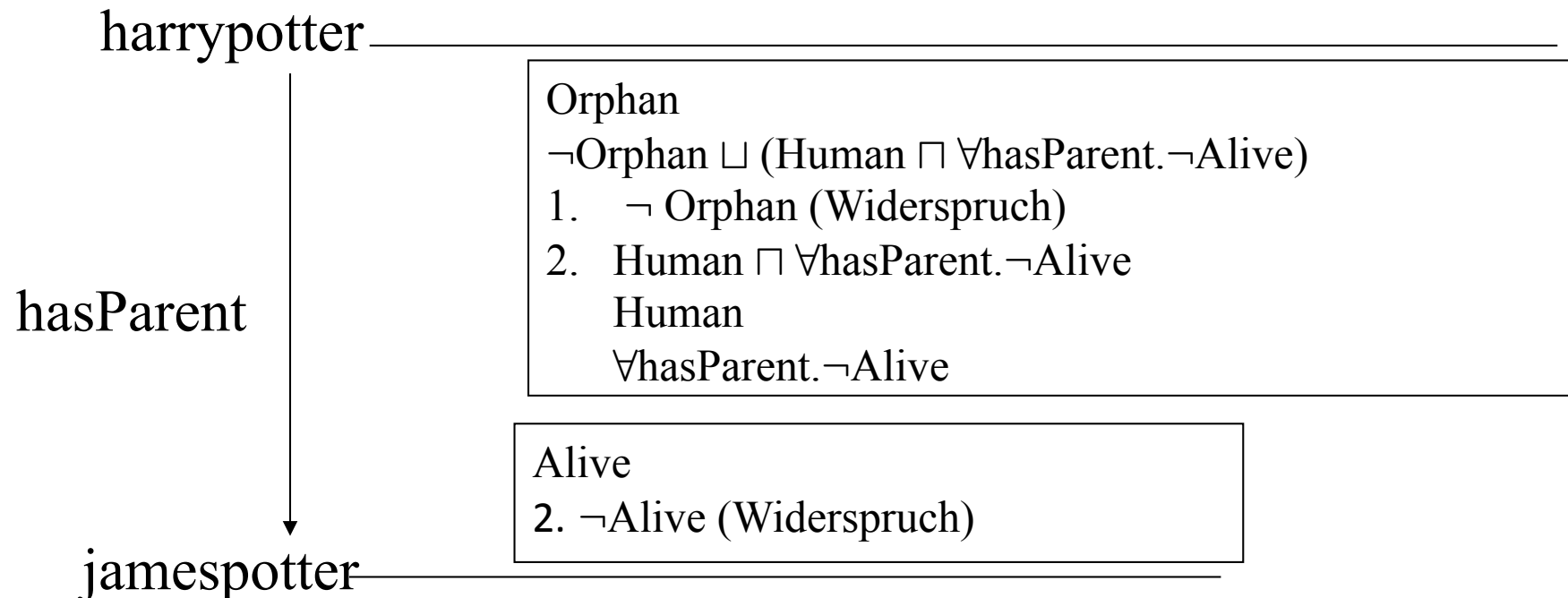
$\neg \text{Alive}(\text{jamespotter})$
d.h. füge hinzu: $\text{Alive}(\text{jamespotter})$
und suche Widerspruch

$\neg \text{Human} \sqcup \exists \text{hasParent.Human}$

$\neg \text{Orphan} \sqcup (\text{Human} \sqcap \forall \text{hasParent.} \neg \text{Alive})$

$\text{Orphan}(\text{harrypotter})$

$\text{hasParent}(\text{harrypotter}, \text{jamespotter})$



Problem mit der Terminierung

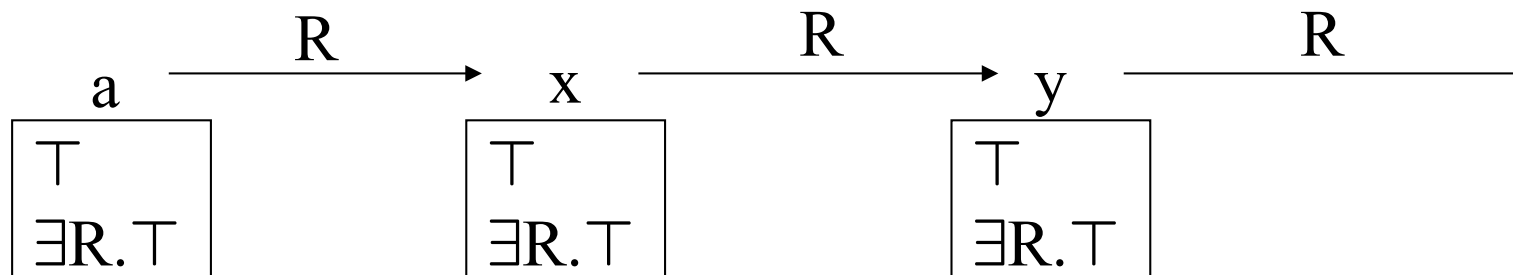
TBox: $\exists R.T$

ABox: $\top(a_1)$

➤ Ist offensichtlich erfüllbar:

Modell M enthält Individuen a_1^M, a_2^M, \dots
und $R^M(a_i^M, a_{i+1}^M)$ für alle $i \geq 1$

➤ aber Tableauverfahren terminiert nicht!

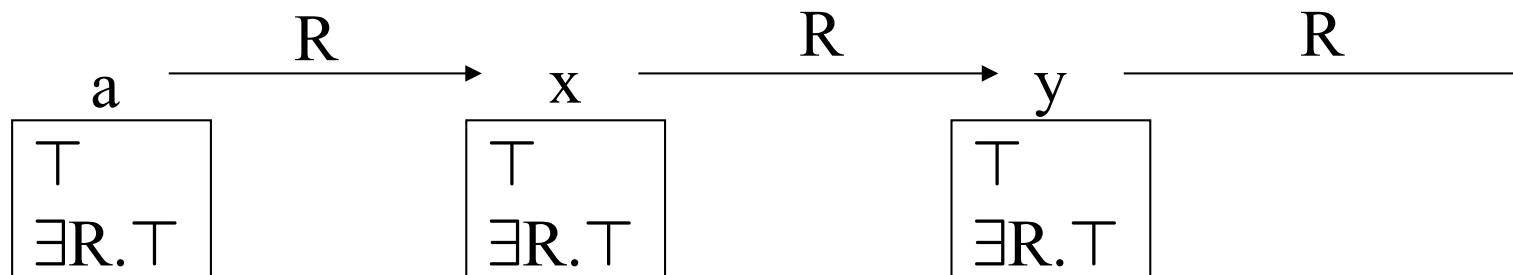


Lösung?

Eigentlich passiert ja nichts neues.

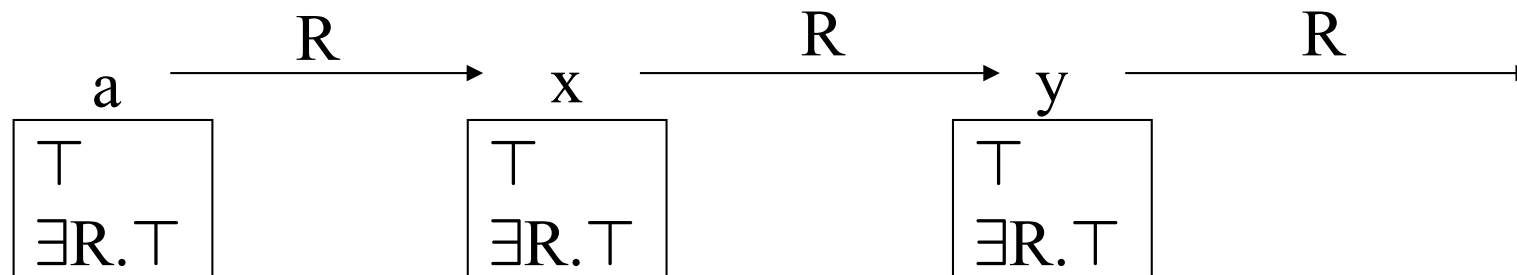
Idee: x muss nicht expandiert werden, da es eigentlich nur eine Kopie von a ist.

⇒ Blocking



Blocking

- x ist *geblockt* (durch y) falls
 - ⇒ y ein Vorgänger von x ist und $L(x) \subseteq L(y)$ ist
 - ⇒ oder ein Vorgänger von x geblockt ist.



x ist in diesem Beispiel durch a geblockt!

Tableau mit Blocking

- Erweitere das Tableau nichtdeterministisch mit den Regeln auf der folgenden Seite, aber **wende eine Regel nur an, falls x nicht geblockt ist!**
- Terminiere, wenn
 - ⇒ die Beschriftung eines Knotens einen Widerspruch enthält (d.h. Klassen C und $\neg C$ enthält) oder
 - ⇒ keine der Regeln mehr anwendbar ist
- Falls das Tableau keinen Widerspruch enthält, ist die Wissensbasis erfüllbar.
 (genauer: sind die Regeln durch geeignete Auswahl so anwendbar, dass kein Widerspruch entsteht und keine Regel mehr anwendbar ist, so ist die Wissensbasis erfüllbar)

Tableauregeln mit Blocking

- \sqcap : Wenn $C \sqcap D \in L(x)$ und $\{C, D\} \not\subseteq L(x)$, dann setze

$$L(x) = L(x) \cup \{C, D\}$$
- \sqcup : Wenn $C \sqcup D \in L(x)$ und $\{C, D\} \cap L(x) = \emptyset$, dann setze

$$L(x) = L(x) \cup \{C\} \text{ oder } L(x) = L(x) \cup \{D\}$$
- \exists : Wenn $\exists R.C \in L(x)$ und x keinen R -Nachfolger y mit $C \in L(y)$ hat, dann
 füge einen neuen Knoten y hinzu mit

$$L(\langle x, y \rangle) = \{R\} \text{ und } L(y) = C$$
- \forall : Wenn $\forall R.C \in L(x)$ und es gibt einen R -Nachfolger y von x mit $C \notin L(y)$, dann setze

$$L(y) = L(y) \cup \{C\}$$
- TBox: Ist C in der TBox und $C \notin L(x)$, dann setze

$$L(x) = L(x) \cup \{C\}$$

Regeln nur anwenden, falls x nicht geblockt ist!

Tableaux-Beweiser

➤ Fact

⇒ <http://www.cs.man.ac.uk/~horrocks/FaCT/>

⇒ SHIQ

➤ Fact++

⇒ <http://owl.man.ac.uk/factplusplus/>

⇒ SHOIQ(D)

➤ Pellet

⇒ <http://www.mindswap.org/2003/pellet/index.shtml>

⇒ SHOIN(D)

➤ RacerPro

⇒ <http://www.sts.tu-harburg.de/~r.f.moeller/racer/>

⇒ SHIQ(D)

Literatur

- Pascal Hitzler, Markus Krötzsch, Sebastian Rudolph, York Sure. **Semantic Web – Grundlagen**. Springer 2008. (ISBN 9783540339939)
- Steffen Staab, Rudi Studer (Editors). **Handbook on Ontologies**. Springer 2003 (ISBN 3540408347).
- Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, Peter Patel-Schneider (eds.), **The Description Logic Handbook**. Cambridge University Press, 2007. (ISBN 9780521781763)